Products : forum question

In [20]:
```
# Put these at the top of every notebook, to get automatic reloading and inline plotting
%reload_ext autoreload
%autoreload 2
%matplotlib inline
```

In [21]:
```
# This file contains all the main external libs we'll use
from fastai.imports import *
from fastai.torch_imports import *
from fastai.transforms import *
from fastai.conv_learner import *
from fastai.model import *
from fastai.dataset import *
from fastai.sgdr import *
from fastai.plots import *
```

In [4]:
```
torch.cuda.is_available()
```

Out[4]:
```
True
```

In [46]:
```
PATH = "data/products/"
sz=150
bs=28
arch=resnet34
```

In [6]:
```
!ls {PATH}
```

```
models  style2.csv  style.csv  style.zip  test  tmp  train
```

In [32]:
```
label_csv = f'{PATH}style2.csv'
n = len(list(open(label_csv)))-1
val_idxs = get_cv_idxs(n)
```

In [33]:
```
n
```

Out[33]: 894

In [34]:
```
len(val_idxs)
```

Out[34]: 178

In [12]:
```
label_df = pd.read_csv(label_csv)
```

In [13]:
```
label_df.head()
```

Out[13]:

	file	product_name
0	0_0_001	shoes
1	0_0_002	shoes
2	0_0_003	shoes
3	0_0_004	shoes
4	0_0_005	shoes

In [14]:
```
label_df.pivot_table(index='product_name', aggfunc=len).sort_values('file' , ascending=False)
```

Out[14]:

product_name	file
handbag	193
shoes	188
ring	88
watches	78
lipstick	66
earrings	63
nail polish	61
boots	60
bracelet	49
necklace	48

In [52]:
```
tfms = tfms_from_model(arch, sz, aug_tfms=transforms_side_on, max_zoom=1.1)
data = ImageClassifierData.from_csv(PATH, 'train', f'{PATH}style2.csv', test_name='test',
                                    val_idxs=val_idxs, suffix='.png', tfms=tfms, bs=bs)
```

In [53]:
```
fn = PATH+data.trn_ds.fnames[100];fn
```

Out[53]:
```
'data/products/train/1_0_010.png'
```

In [54]:
```
img = PIL.Image.open(fn); img
```

Out[54]:

In [22]:
```
img.size
```

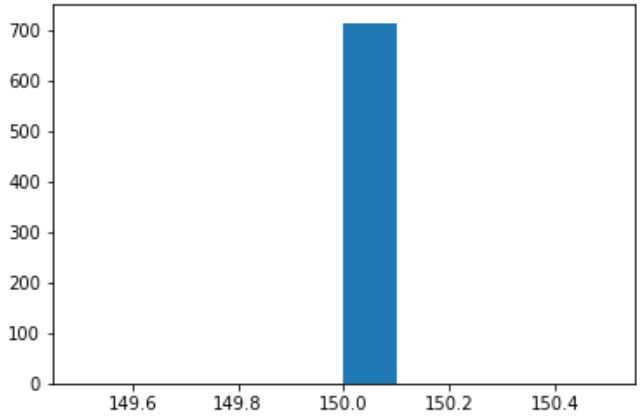
Out[22]:
```
(150, 150)
```

In [27]:
```
size_d = {k: PIL.Image.open(PATH+k).size for k in data.trn_ds.fnames}
row_sz,col_sz = list(zip(*size_d.values()))
row_sz=np.array(row_sz); col_sz=np.array(col_sz)
row_sz[:5]
```

Out[27]:
```
array([150, 150, 150, 150, 150])
```

In [29]:
```
plt.hist(col_sz)
```

Out[29]:
```
(array([  0.,   0.,   0.,   0.,   0., 715.,   0.,   0.,   0.,   0.]),
 array([149.5, 149.6, 149.7, 149.8, 149.9, 150. , 150.1, 150.2, 150.3, 150.4, 150.5]),
 <a list of 10 Patch objects>)
```



In [37]:
```
len(data.trn_ds)
```

Out[37]:
```
716
```

In [38]:
```
len(data.classes)
```

Out[38]:
```
11
```

In [39]:
```
data.classes[:5]
```

Out[39]:
```
['boots', 'bracelet', 'earrings', 'handbag', 'lipstick']
```

In [55]:
```
learn = ConvLearner.pretrained(arch, data, precompute=True)
```

```
100%|██████████| 1/1 [00:00<00:00, 28.30it/s]
```

In [56]:
```
learn.fit(0.2, 2)
```

Failed to display Jupyter Widget of type `HBox`.

If you're reading this message in the Jupyter Notebook or JupyterLab Notebook, it may mean that the widgets JavaScript is still loading. If this message persists, it likely means that the widgets JavaScript library is either not installed or not enabled. See the Jupyter Widgets Documentation (https://ipywidgets.readthedocs.io/en/stable/user_install.html) for setup instructions.

If you're reading this message in another frontend (for example, a static rendering on GitHub or NBViewer (https://nbviewer.jupyter.org/)), it may mean that your frontend doesn't currently support widgets.

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-56-8751e61b7595> in <module>()
----> 1 learn.fit(0.2, 2)

~/fastai/courses/dl1/fastai/learner.py in fit(self, lrs, n_cycle, wds, **kwargs)
    207         self.sched = None
    208         layer_opt = self.get_layer_opt(lrs, wds)
--> 209         return self.fit_gen(self.model, self.data, layer_opt, n_cycle, **kwargs)
    210
    211     def warm_up(self, lr, wds=None):

~/fastai/courses/dl1/fastai/learner.py in fit_gen(self, model, data, layer_opt, n_cycle, cycle_len, cycle_mult, cycle
_save_name, use_clr, metrics, callbacks, use_wd_sched, norm_wds, wds_sched_mult, **kwargs)
    154         n_epoch = sum_geom(cycle_len if cycle_len else 1, cycle_mult, n_cycle)
    155         return fit(model, data, n_epoch, layer_opt.opt, self.crit,
--> 156             metrics=metrics, callbacks=callbacks, reg_fn=self.reg_fn, clip=self.clip, **kwargs)
    157
    158     def get_layer_groups(self): return self.models.get_layer_groups()

~/fastai/courses/dl1/fastai/model.py in fit(model, data, epochs, opt, crit, metrics, callbacks, **kwargs)
    104             i += 1
    105
--> 106         vals = validate(stepper, data.val_dl, metrics)
    107         if epoch == 0: print(layout.format(*names))
    108         print_stats(epoch, [debias_loss] + vals)

~/fastai/courses/dl1/fastai/model.py in validate(stepper, dl, metrics)
```

In [ ]: